Problem 1. Determine the truth of the following statements about counting sort for a universe $U$.

Counting sort runs in $O(n + |U|^2)$ time. □ True □ False

For valid universes, counting sort is slower than mergesort on all inputs. □ True □ False

For valid universes, counting sort is faster than mergesort on all inputs. □ True □ False

For some inputs, counting sort runs in $\Theta(1)$ time. □ True □ False

Problem 2. Indicate whether counting sort can (in theory) be implemented in C++ to sort the following types.

chars □ Yes □ No

floats □ Yes □ No

strings □ Yes □ No

ints □ Yes □ No
Problem 3. Complete the following implementation of counting sort.

```c
void sort(char* A, int n)
{
    int H[ _____ ];

    for (int i = 0; i < _____; ++i)
        H[i] = _____;

    for (int i = 0; i < n; ++i)
        H[A[i]] += _____;

    int i = 0;
    int h = 0;
    while (h < _____ )
    {
        if (H[h] == _____)
            ++h;
        else
        {
            A[i] = _____;
            ++i;
            H[h] = H[h] _____ 1;
        }
    }
}
```
**Problem 4.** Determine the truth of the following statements about stable sorting.

Every implementation of counting sort is stable.  □ True   □ False

The following sorting algorithm is stable:
```c
void sort(float* A, int n)
{
    for (int r = 0; r < n; +r)
        for (int i = 0; i < n-1; ++i)
            if (A[i] > A[i+1])
                swap(A[i], A[i+1]);
}
```
□ True   □ False

Mergesort has a stable implementation.  □ True   □ False

The following sorting algorithm is stable:
```c
void sort(float* A, int n)
{
    for (int r = 0; r < n; +r)
        for (int i = 0; i < n-1; ++i)
            if (A[i] > A[i+1])
                swap(A[i], A[i+1]);
}
```